Programação Orientada a Objetos
Aula 01

João Choma Neto

joao.choma@gmail.com

O que veremos nesse semestre?

- Introdução a linguagem de programação
- Orientação a Objetos

O que veremos nesse semestre?

- Relação entre UML e orientação a objetos
- Utilização de IDEs

O que veremos nesse semestre?

- Linguagem Java
- Princípios de orientação a objeto

O que veremos nesse semestre?

- Interface
- Persistência de dados

Referências utilizadas

- ASCENCIO, A. F. G. Fundamentos da programação de computadores. 2ª. Edição. São Paulo: Pearson Prentice Hall.

- https://www.oracle.com/

Avaliação

- 02 provas escritas
- 02 trabalhos práticos

Paradigmas de Programação

Paradigma de programação

- Um paradigma pode ser entendido como a **forma** com a qual se decide **resolver** determinado **problema** por meio da programação

Linguagem de programação

- Programação imperativa
- Paradigma de programação mais comum
- Utiliza instruções que modificam o estado de variáveis para controlar o fluxo de execução do programa

- Programação imperativa

```c
#include <stdio.h>

int main() {
    int x = 5;

    if (x > 0) {
        printf("x é positivo");
    } else {
        printf("x é não-positivo");
    }

    return 0;
}
```

Linguagem de programação

- Programação funcional
- Paradigma de programação se concentra no uso de funções matemáticas
- Utiliza funções para transformar dados

- Programação funcional

```
; Definir uma função para calcular o fatorial de um número
(define (fatorial n)
  (if (<= n 1)
      1
      (* n (fatorial (- n 1)))))

; Definir uma função para calcular a soma dos elementos de uma lista
(define (soma-lista lst)
  (cond
    ((null? lst) 0)
    (else (+ (car lst) (soma-lista (cdr lst))))))

; Definir uma função para calcular a sequência de Fibonacci
(define (fibonacci n)
  (cond
    ((= n 0) 0)
    ((= n 1) 1)
    (else (+ (fibonacci (- n 1)) (fibonacci (- n 2))))))

; Exemplo de uso das funções
(displayln (fatorial 5)) ; Imprime "120"
(displayln (soma-lista '(1 2 3 4 5))) ; Imprime "15"
(displayln (fibonacci 10)) ; Imprime "55"
```

Linguagem de programação

- Programação baseada em lógica
- Paradigma de programação usa a lógica matemática para resolver problemas de programação
- Baseado em regras e fatos lógicos

- Programação funcional

```
ir fatos para relacionar algumas cores com alguma
e(joao, laranja).
e(maria, banana).
e(pedro, abacaxi).
e(joana, abacaxi).
e(paulo, laranja).
e(ana, uva).

ir regras para relacionar pessoas com suas frutas
referida(Pessoa, Fruta) :-
ta_de(Pessoa, Fruta),
(gosta_de(outra_pessoa, Fruta)),
(gosta_de(Pessoa, outra_fruta)).

ir uma consulta para encontrar a fruta preferida
a_preferida(joao, Fruta).
```

Programação Concorrente

- Programação concorrente
- Paradigma de programação se concentra na execução simultânea de várias tarefas ou processos

- Programação concorrente

```
port java.util.concurrent.ExecutorService;
port java.util.concurrent.Executors;

blic class ExemploConcorrente {

  public static void main(String[] args) {
      // Criar um pool de threads com duas threads
      ExecutorService executor = Executors.newFixedThre

      // Executar duas tarefas concorrentemente
      executor.submit(new TarefaConcorrente("Tarefa 1")
      executor.submit(new TarefaConcorrente("Tarefa 2")

      // Encerrar o pool de threads quando as tarefas e
      executor.shutdown();
  }
```

- Programação concorrente

```
class TarefaConcorrente implements Runnable {

    private String nome;

    public TarefaConcorrente(String nome) {
        this.nome = nome;
    }

    public void run() {
        System.out.println("Iniciando " + nome);
        try {
            // Dormir por um período aleatório de tempo para
            Thread.sleep((long) (Math.random() * 5000));
        } catch (InterruptedException e) {
            System.out.println("Interrupção durante " + nome
        }
        System.out.println("Concluindo " + nome);
    }

}
```

Linguagem de programação

- Programação orientada a objetos (POO)
- Paradigma de programação enfatiza a organização do código em classes e objetos
- A tarefas são realizadas pela interação dos objetos
- Abstração do mundo real em objetos

- Programação orientada a objetos

```
ic class Pessoa {

// Atributos da classe Pessoa
private String nome;
private int idade;

// Construtor da classe Pessoa
public Pessoa(String nome, int idad
    this.nome = nome;
    this.idade = idade;
}
```

- Programação orientada a objetos

```
; getter e setter para acessar os atributos da cla
ring getNome() {
 nome;


d setNome(String nome) {
ome = nome;


 getIdade() {
 idade;


d setIdade(int idade) {
dade = idade;
```

- Programação orientada a objetos

```
in para testar a classe Pessoa
c void main(String[] args) {
 uma nova instância da classe Pessoa
essoa1 = new Pessoa("João", 30);

r o método saudar da instância pessoa1
saudar();

ar o nome e idade da instância pessoa1 usando
setNome("Maria");
setIdade(25);

r o método saudar da instância pessoa1 novamer
saudar();
```

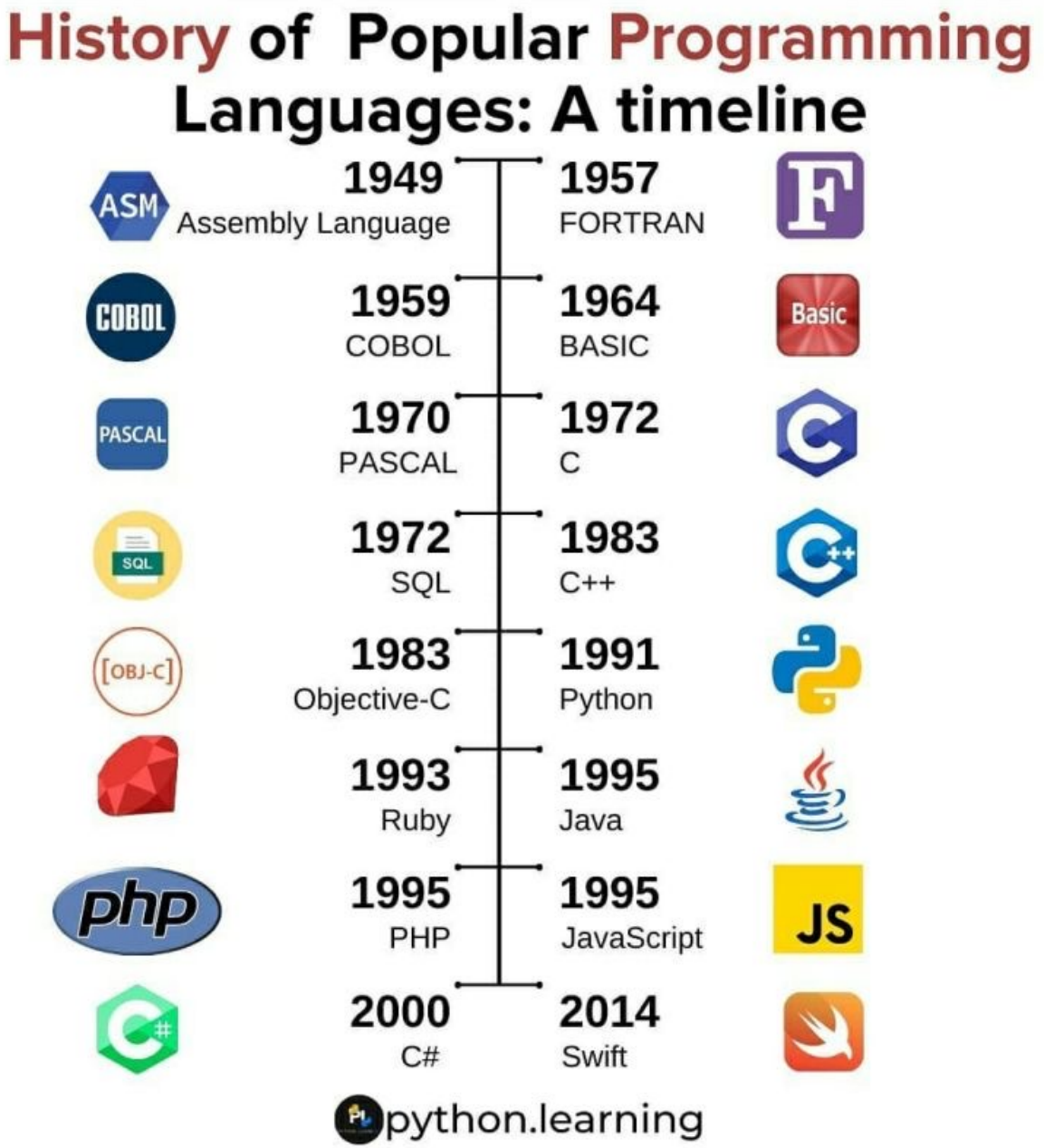
Primeiras linguagens de programação

- Fortran
- Lisp
- Algol
- Smalltalk
- C

Linguagens mais utilizadas atualmente

Classificação	Linguagem de programação	Quota de mercado	Tendência
1	Python	27.61%	-2.8%
2	Java	17.64%	-0.7 %
3	JavaScript	9.21%	+0.4 %
4	C#	7.79%	+0.8 %
5	C/C++	7.01%	+0.4 %

python.learning • Seguir



Programação Orientada a Objetos

POO

- O paradigma da POO (Programação Orientada a Objetos) é um modelo de análise, projeto e programação baseado na aproximação entre o mundo real e o mundo virtual

- Criação e interação entre objetos

POO

- A primeira linguagem de programação com paradigma de orientação a objetos foi criada em 1970

- Criado por Alan Kay

POO

- Smalltalk

- Primeira linguagem a usar conceitos de classes, objetos, atributos e métodos.

POO – Abstração do mundo real

- Modelar o comportamento de entidades do mundo real como objetos em software
- Os objetos são entidades que possuem valores/pesos/dados e comportamentos

POO – Vantagens

- Confiável - qualquer intervenção que seja necessária não afetará outros pontos do sistema
- Ajustável – a herança garante que outras partes que utilizam uma classe sejam beneficiadas
- Extensível – reutilização de código

POO – Vantagens

- Reutilizável – um mesmo objeto pode ser utilizado em diferentes sistemas (objeto cliente)
- Natural – abstração do mundo real

POO - Base

- Encapsulamento
- O encapsulamento é a capacidade que determinado método ou atributo de um objeto tem de se manter invisível
- É aquele famoso pensamento de saber o que faz, mas não saber como se faz

POO - Base

- Herança
- Trata-se de uma relação de receber algo pré-existente
- Classe Mãe doa seu funcionamento a classe filha

POO - Base

- Polimorfismo
- Um mesmo método pode ser utilizado em diferentes objetos, de diferentes classes.

POO - Base

- Abstração
- A ideia principal é representar um objeto de forma abstrata
- Uma classe abstrata contem atributos e métodos
- Sua implementação é de responsabilidade da classe que herda a classe abstrata

Linguagem Trabalhada Java

Linguagem Java

- A linguagem Java carrega heranças de muitas linguagens
- Algol
- CPL
- B
- C
- C++

Origem da linguagem Java

- 1991
- Parecida com C++
- Fortemente tipada
- Portável – Roda em uma máquina virtual

- Recebeu o nome de Java em 1995

Vamos praticar

Atividade

- Instalar o máquina virtual Java
- Instalar a IDE para codificação do Java

Links

- https://netbeans.apache.org/download/nb17/index.html
- https://www.oracle.com/br/java/technologies/downloads/#jdk19-windows

Exercícios para praticar

System.out.println("hello word");

Scanner scanner = new Scanner(System.in);

Exercícios para praticar

1. Faça um programa em Java que imprima a mensagem "Hello World!" na tela.
2. Faça um programa que peça ao usuário que digite um número inteiro e, em seguida, imprima na tela se esse número é par ou ímpar.
3. Faça um programa que leia um número inteiro e imprima todos os números de 1 até esse número.

Programação Orientada a Objetos
Aula 01

João Choma Neto

joao.choma@gmail.com